The analysis ecosystem

~~My vision~~

My remarks

Eduardo Rodrigues
University of Cincinnati

HSF HEP analysis ecosystem workshop

Amsterdam, The Netherlands, 22-4 May 2017

Introduction – disclaimer(s)

This "my vision talk" IS

- From a physicist with great interest in analysis software/tools
- Opinionated, hence biased and non-comprehensive
- Intended to trigger discussion

(Many key points got raised already yesterday, unsurprisingly)

It is NOT

- Meant as publicity against anything

On the worst ennemies of analysts

1. **Lack of knowledge and/or incompetence**
   Solution: "You might want to look into …", a British would say
2. **Inertia – the thinking "This is not working well but I will keep working this way rather than spending time investigating (potentially rewarding) alternatives!"**
   Solution: the hard-to-trigger shift in culture (problem)
3. **Inappropriate and/or not timely available software working environment**
   - Broad subject touching many points discussed here (SW stacks, notebooks, etc.)
   Solution: I rely on a great core software team
4. **Inappropriate analysis software ecosystem**
   Solution: We are here to discuss this … ;-)
5. **Bad distributed computing model - user interaction with the Grid just isn't great, let's be honest**
   - For example LHCb has Ganga, very much underpowered and often problematic
   Solution: we probably need a common tool (sustainability & maintainability ✓)
6. **Amount of data and the complexity of analyses (tasks)**
   - But this should actually be seen as a good thing ;-)
   Solution? I do not want one. I rather want the "problem" of more data!

On the analysis ecosystem and analysis model(s)

(R)Evolution of the analysis ecosystem:

Over a decade ago HEP caught up with a TGV train called “Python” …
… but we paid too little attention to the bold Shinkansen called “Machine learning” !

It took us until the 1st years of the LHC data taking era to realise what we missed, or have been missing, and a few more years to react accordingly !

⇒ Let’s avoid repeating this mistake again …

- **The analysis ecosystem needs to be in phase with the analysis model(s)**
  (said already yesterday in LHCb presentation but repeat it since very important)
- **Real-time analyses such as “Turbo analyses” in LHCb make the issue ever more important**
  - This is the future and other experiments should seriously brainstorm on how to take such a route within their constraints and alike

Running jobs – connection to Grid/Cloud computing

It will be impossible in 3-5 years from now to analyse data the way we do it today !
- Certainly true for LHCb

What kind of jobs do analysts submit ?

- **On data: skimming&co tasks, complex fits**
- **On simulated data: analysis jobs similar to those performed on data**
- **On calibration samples: data-driven studies to correct or determine unbiased efficiencies, etc.**
- **Toy studies: typically (very) large sets to validate fits**

In 3-5 years

- **Tools are not performant enough for tasks to be run in a reasonable amount of time**
- **Keep also in mind that size of samples to run over will increase with data size**
  - **Charm analyses in LHCb: this may mean gigantic samples**
- **We either need new tool(s) or stronger development and support. Or both**
  - **A common tool (at least for the LHC) would be desirable, judging from experience in LHCb**

ROOT vs non-HEP software (Am not against ROOT. Use it every-ish day.)

- ROOT cannot and should not try to compete with dedicated non-HEP tools
  - Especially true for machine learning
- Though I'm sure "we" will still do it …
  - Reason is, physicists like to reinvent the wheel unlike engineers

- At the same time ROOT is central to experiments – no data taking without it
- Try to decouple as much as possible and focus on what we do best and need most

- Paramount to expose ROOT in Python. Clear from usage and interoperability with non-HEP
- ROOT7 should simply support Python 3
  - Let's finally make the step-function move, looking forward to the future

- "Toolset" vs "toolkit" [Am no native speaker so allow me to play a bit with the language ;-)]
  - ROOT more of a toolkit whereas the Python scientific ecosystem is a toolset
- Toolsets are the future – need is highly correlated with modularity

Working environments & frameworks

Remember: physicists (not the only ones) hate it when things do not work out of the box !

- **SWAN & ROOT notebooks are a very nice initiative following a trend seen outside HEP**
- **Find it an excellent tool for small interactive development , small analysis tasks, things such as tutorials and outreach**
  - **It will be very interesting to see how Belle II evolves with their analysis environment with jupyter + basf2, what their experience will be there**
- **Doubt it will be suitable as a daily full working environment**
- **BTW this route requires thought on how to interact with the Grid (well and efficiently) from the notebook, I think**

- **We will be working on VMs most likely, and containers are being increasingly used**
  - **Storage of data and/or user files (CERNBox) and software (CVMFS) are paramount**
- **Need more streamlined/adequate/adapted strategies here**
- **Would personally welcome more teaching to analysists on these topics …**

Other initiatives ?

Worth mentioning

Everware

- **Edit & run someone else's code even if code has complicated instructions**
- **Encourage reuse of software between researches**
- **Built in Python, uses jupyter and Docker**
- **Could be used e.g. for tutorials, outreach at schools !**
- **See http://everware.xyz/ …**

Analysis ecosystem landscape (not comprehensive!)

(Recap from yesterday.)

Purpose	Software	Language of use	HEP ?
Data manipulation	ROOT	C++ & Python	Yes
	numpy, pandas, bcolz	Python	No
	root_numpy, root_pandas	Python	Yes
Machine learning (classification, regression)	TMVA	C++ & Python	Yes
	scikit-learn, TensorFlow	Python	No
	Keras, XGBoost, etc.	C++	No
	NeuroBayes	C++	No
Plotting	ROOT	C++ & Python	Yes
	matplotlib, seaborn, bokeh	Python	No
Fitting	RooFit	C++ & Python	Yes
	<Institute/user packages>	C++	Yes
Statistics	CLs	Python	Yes
	RooStats	C++ & Python	Yes
Reweighting	hep_ml	Python	Yes & no
	Meerkat	C++ & Python	Yes
Error propagation	uncertainties, mcerp	Python	No

Tools are largely there. Need to focus more on interoperability …

On software sustainability

- **Clearly an important aspect/requirement !**

We often see little single-person(-ish) projects/packages,
which evolve like quantum fluctuations and often have rather short lifetimes

- **We need "production out of the vacuum", with energy above threshold and the possibility to survive the lifespan of an experiment, and more**
- **This requires reasonably-sized teams, collaboration**

- **Community-driven and community-oriented projects tick the points of collaborative work (funding agencies love it!), sustainability, chances of survival, etc.**

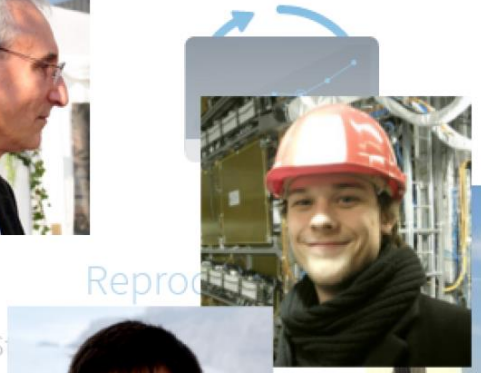
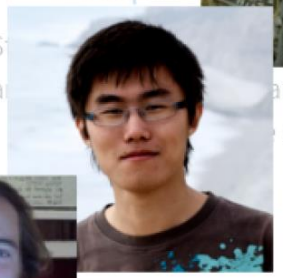
This is what makes projects like DIANA/HEP relevant & important !






Collaborative

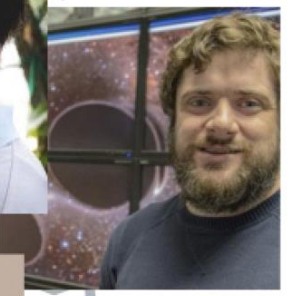
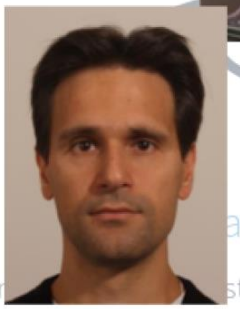
Establish infrastructure f… collaborative analysis, build… ful patterns used for the Higgs … nd enabling a deeper commun… e theoretical community and the experiment


Reproc…

S… and dat… concep…



…rability

…prove the interoperability of HEP tools with … software ecosystem, …actices and algorithms …iplines into HEP

Faster Processing

Increase the CPU and IO performance needed to reduce the iteration time so crucial to exploring new ideas

…re

Deve… exploit emerging many- and multi-core hardware.
Promote the concept of software as a research product.

…aining

Pr… students in all of our core research topics.

Collaborative Analyses

Establish infrastructure for a higher-level of collaborative analysis, building on the successful patterns used for the Higgs boson discovery and enabling a deeper communication between the theoretical community and the experimental community

Reproducible Analyses

Streamline efforts associated to reproducibility, analysis preservation, and data preservation by making these native concepts in the tools

Interoperability

Improve the interoperability of HEP tools with the larger scientific software ecosystem, incorporating best practices and algorithms from other disciplines into HEP

Faster Processing

Increase the CPU and IO performance needed to reduce the iteration time so crucial to exploring new ideas

Better Software

Develop software to effectively exploit emerging many- and multi-core hardware.
Promote the concept of software as a research product.

Training

Provide training for students in all of our core research topics.



Software products

- DIANA/HEP organization on GitHub
- Carl: a toolbox for likelihood-free inference (documentation, code) DOI 10.5281/zenodo.47798
- Histogrammar: a multilingual specification for histogram aggregation with implementations in Python and Scala - docs, GitHub
- Femtocode: a query language, execution engine, and query server for complex data (in development) - GitHub
- Scikit-HEP: a community-driven Python project for HEP core and common tools (web page, GitHub)
- Scikit-Optimize: a library for sequential model-based optimization (documentation, code)
- yadage: a language and engine for flexible, distributed workflows - docs code
- REANA: a system that permits to instantiate research data analyses on the cloud. docs code



Faster Processing

Increase the CPU and IO performance needed to reduce the iteration time so crucial to exploring new ideas

Better Software

Develop software to effectively exploit emerging many- and multi-core hardware.
Promote the concept of software as a research product.

Training

Provide training for students in all of our core research topics.

The Scikit-HEP project

The idea, in just one sentence

The Scikit-HEP project (http://scikit-hep.org/) is a community-driven and community-oriented project with the aim of providing Particle Physics at large with a Python package containing core and common tools.

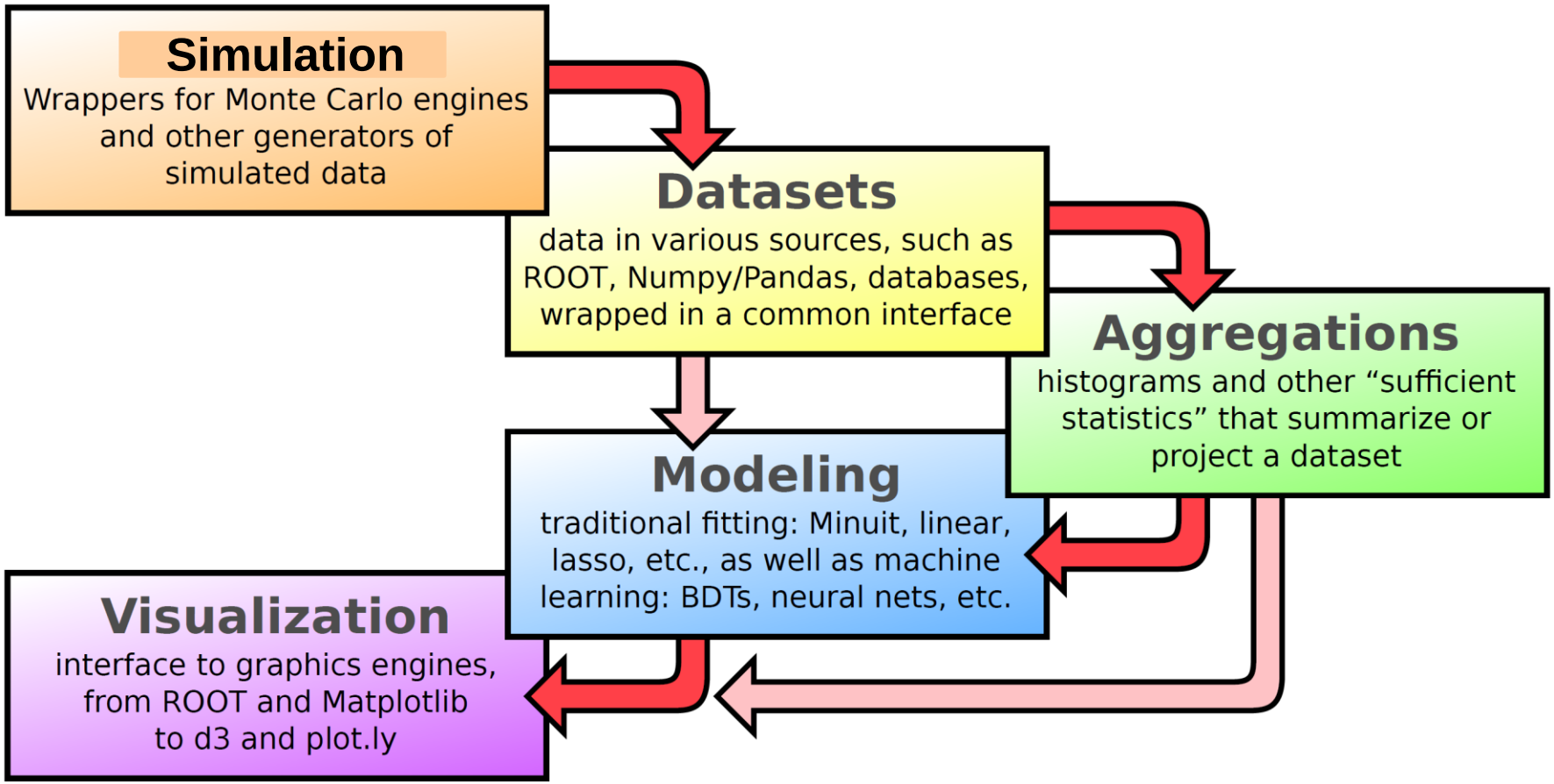
What it is NOT …

- **A replacement for ROOT**
- **A replacement for the Python ecosystem based on NumPy, scikit-learn & co.**

… and what IT IS

- **Bridge/glue between the ROOT-based and the Python scientific ecosystem**
  - **Expand typical toolkit of HEP physicists**
  - **Common definitions and APIs to ease "cross-talk"**
- **Project similar to the Astropy project – learn from good examples ;-)**
- **We are building a community, engaging with (future) collaborators in various experiments**

The Scikit-HEP project – 5 « pillars »



They cover all grand topics … !

The Scikit-HEP software package (non-exhaustive!)

- **Dataset**
  - Common interface for data in various sources
  - Dealing with ROOT TTree and Numpy arrays for a start (profit from root_numpy project!)
- **Aggregation**
  - Summarise or project a dataset
  - Typically data aggregation = histogram
  - Make use of the Histogrammar project?
- **Modeling**
  - Data models and fitting utilities
  - Will need careful design to talk smoothly to the Python scientific ecosystem at large
- **Visualization**
  - Interface to graphics engines such as ROOT and matplotlib, among others
  - Build from rootpy project!
- **Simulation**
  - utilities, wrappers for Monte Carlo engines and other generators of simulated data
- Modules for **units** and **constants**
- **Maths** and **statistics tools**

Affiliated packages

- Take good concept from Astropy of an *affiliated package*:
  Python package not part of the Scikit-HEP core but related to, and seen as part of, the Scikit-HEP community and project

- Allows expansion of toolkit avoiding a gigantic do-everything package

- Bring-in functionality specific to certain topics/areas not of the widest community interest

- Potential examples are
  - root_numpy – already affiliated and under the Scikit-HEP community "hat"
  - Hydra, specifically a Python API to this header-only C++ library for data analysis in massively parallel platforms
  - hep_ml, a ML library with miscellaneous tools for HEP
  - Many more.

Thank you